Discover Applied Sciences

Research

Human–machine interaction and implementation on the upper extremities of a humanoid robot

Panchanand Jha[1] · G. Praveen Kumar Yadav[2] · Din Bandhu[3] · Nuthalapati Hemalatha[4] · Ravi Kumar Mandava[5] · Mehmet Şükrü Adin[6] · Kuldeep K. Saxena[7] · Mahaboob Patel[8]

Received: 19 July 2023 / Accepted: 13 February 2024
Published online: 20 March 2024

Abstract
Estimation and tracking the various joints of the human body in a dynamic environment plays a crucial role and it is a challenging task. Based on human–machine interaction, in the current research work the authors attempted to explore the real-time positioning of a humanoid arm using a human pose estimation framework. Kinect depth sensor and media pipe framework are used to obtain the three-dimensional position information of human skeleton joints. Further, the obtained joint coordinates are used to calculate the joint angles using the inverse kinematics approach. These joint angles are helpful in controlling the movement of the neck, shoulder, and elbow of a humanoid robot by using Python-Arduino serial communication. Finally, a comparison study was conducted between the Kinect, MediaPipe, and real-time robots while obtaining the joint angles. It has been found that the obtained result from the MediaPipe framework yields a minimum standard error compared to Kinect-based joint angles.

Article Highlights

- Development of a real-time framework for obtaining various joint postures of the humanoid arm by using a Kinect depth sensor and Media pipe framework
- Implementation of inverse kinematics approach for obtaining various joint angles of the humanoid arm
- Standard error calculation between the joint angles obtained from inverse kinematics (that is, robot joint angles), the Kinect depth sensor, and the Media framework.

Keywords Humanoid · Dynamic environment · Kinect · Media pipe · Inverse kinematics

✉ Din Bandhu, din.bandhu@manipal.edu; ✉ Mahaboob Patel, mahaboob.patel@wsu.edu.et | [1]Department of Mechanical Engineering, Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India. [2]Department of Mechanical Engineering, G. Pulla Reddy Engineering College, Kurnool, Andhra Pradesh 518007, India. [3]Department of Mechanical and Industrial Engineering, Manipal Institute of Technology Bengaluru, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India. [4]Department of Electronics and Communication Engineering, R.V.R & J.C College of Engineering, Guntur, Andhra Pradesh, India. [5]Department of Mechanical Engineering, Indian Institute of Information Technology Design and Manufacturing, Kurnool, Andhra Pradesh 518008, India. [6]Besiri OSB Vocational School, Batman University, Batman, Turkey. [7]Division of Research and Development, Lovely Professional University, Phagwara, India. [8]Department of Mechanical Engineering, College of Engineering, Wolaita Sodo University, Soddo, Ethiopia.

1 Introduction

In recent years, research on humanoid robots is getting more and more attention due to their versatile applications such as assisting elderly or physically challenged people, healthcare, public entertainment, personal care, education, search-rescue operations, and manufacturing. It has been observed that many humanoid robots are mimicking human behavior such as walking, talking, grasping, etc. Based on the above applications researchers started to develop humanoid robots and developed the first humanoid robot in 1930 in the USA. Later on, in 1966, Waseda University developed a humanoid robot that is, WABIAN-II, and in 1996, Honda Corporation developed a humanoid robot known as ASIMO. Due to the usage of humanoid robots in various fields, many organizations such as Toyota, Samsung, Hanson Robotics, NASA, Boston Dynamics, MIT, UBTECH, Columbia University, etc. developed various versions of the humanoid robot [1, 2]. Many humanoid robots are capable of intelligent behavior due to recent advancements in artificial intelligence (AI), machine learning (ML), computer vision, cognitive computing, natural language processing, and accelerated hardware. The above-said techniques are helpful in extracting useful information from its environment through sensors. Computer vision and artificial intelligence provide a new perspective to humanoid robots for their basic actions like walking and grasp manipulation. Moreover, the application of computer vision to contextualize, visualize, and react to their environment can be predominant. It has been made that computer vision techniques are the building blocks for image and video processing. It is mainly concerned with object detection, image processing, gesture recognition, image segmentation, object tracking, and pose estimation. One of the most important tasks is to estimate the human pose and track the various landmarks (joint locations). Human pose estimation (HPE) predicts and classifies the posture of the human body and its joint locations in an image or video format. The capturing method of 2D/3D joint coordinates of the shoulder, elbow, wrist, knees, ankles, arms, eyes, and ears, are the key points to describe the pose of a human. There are two main categories of pose estimation techniques, (i) 2D pose estimation: this extracts the x and y coordinates of joint location for all joint landmarks. (ii) 3D pose estimation: this extracts z-coordinates or depth information along with (x, y) coordinates. This pose estimation can be further categorized as kinematic-based, shape or contour-based, and volume-based models [3–8]. Many researchers are using skeleton tracking algorithms that can be based on the classical approach [9] as well as intelligent approaches [10–14]. It has been observed that researchers around the world are using deep CNN architectures for human pose estimations, some of them are listed in Table 1. Bujalance and Moutarde [15] presented a real-time control of the universal robot arm using a pose estimation framework. In this work, the authors adopted the open pose and human mesh recovery (HMR) frameworks. Later on, they calculated the inverse kinematics (IK) and forward kinematics (FK) to calculate the joint angles from the given pose key points. Chamorro et al. [16] proposed a lidar-based gesture recognition system to control the mobile robot for teleoperation. The authors adopt the long short-term memory (LSTM) and CNN architecture for pose estimation. The proposed work uses static and dynamic input from the lidar and with the help of Euclidean clustering initial pose is extracted from a point cloud. Zimmermann et al. [17] presented a human pose estimation framework using the open pose library and Voxel Pose Net. The adopted Voxel Pose Net is inspired by U-Net or also known as encoder-decoder neural network architecture. In this work, the PR2 robot is used to imitate the action of actors using the pose estimation framework and compared with marker-based estimation techniques. Gago et al. [18] discussed the application of the LSTM network to convert the natural language into Spanish sign language. To understand sign language, the authors used human skeleton or pose estimation. Therefore, they adopted open pose and the skeleton retriever library for further acquisition of joints. Finally, these sign languages are tested on the TEO humanoid robot. Amini et al. [19] proposed a novel deep-learning model for the 2D pose estimation of a humanoid robot. In the current research work, the authors introduced a humanoid robot pose dataset and the current model is working based on the bottom-up single-stage encoder-decoder architecture. It is an efficient algorithm when compared top-down approaches. Further, a comparative study has been made with other states-of-art models. Michel et al. [20] presented a marker less 3D human pose estimation method for tracking joint locations. The authors adopted three different approaches namely OpenNI, HYBRID, and FHBT. A 3D human pose estimation is used for positioning of NAO robot arm and conducted a comparative study for all three adopted methods. Later on, Liang et al. [21] proposed a vision-based marker less pose estimation framework for articulated construction robots. The authors used, a stacked hourglass deep neural network to estimate the joint locations for an articulated robot. The concept is similar to human pose estimation but it has been used to extract the joint information of articulated robots. Cai et al. [22] discussed a patient's upper limb motion tracking using a Kinect depth camera with VICON markers. The Barret WAM manipulator is used to track the patient's upper limb movement for the rehabilitation exercise. Later on, Kinect v2 with VICON markers is used to extract the pose information. Finally, qualitative analysis has been made on joint angles and velocities.

Table 1 Human pose estimation frameworks

S. No.	Authors	Title	Method/framework	HPE	Landmarks	Dataset	Year
1	Wan et al. [34]	TSNet: Tree structure network for human pose estimation	Based on Heatmap, Convolution module, Hourglass module	Whole body	14	Human Pose (MPII) datasets	2022
2	Zhang et al. [35]	AdaFuse: Adaptive multiview fusion for accurate human pose estimation in the wild	AdaFuse: Heatmaps, fully connected CNN	Whole body	12	Human3.6 M, Total Capture and CMU Panoptic	2021
3	Bashirov et al. [28]	Real-time RGBD-based extended body pose estimation	MLPNN	Whole body	14	Motion capture AMASS dataset	2021
4	Eusanio et al. [36]	RefiNet: 3D Human Pose Refinement with Depth Maps	Deep CNN	Whole body	14	Baracca, a novel dataset acquired with a set of RGB, depth, and thermal cameras. It contains nearly 10 k frames of 30 different subjects from 8 different points of view	2021
5	Miki et al. [37]	Robust human pose estimation from distorted wide-angle images through iterative search of transformation parameters	CNN	Whole body	14	MPII datasets	2020
6	Miura et al. [38]	3D human pose estimation model using location maps for distorted and disconnected images by a wearable omnidirectional camera	High-resolution network (HRNet) CNN	Whole body	12	RGB-D sensor dataset	2020
7	Chen et al. [39]	Nonparametric structure regularization machine for 2D hand pose estimation	Nonparametric Structure Regularization Machine (NSRM) with VGG-19 backbone	Hand	20	OneHand 10 k and CMU Panoptic Hand dataset	2020
8	Valentin et al. [33]	BlazePose: On-device Real-time Body Pose tracking	combined heatmap, offset, and regression approach with regression encoder network	Whole body	33	Custom dataset, In-the-wild dataset, in-house collected gesture dataset, and Synthetic dataset	2020
9	D'Eusanio et al. [40]	Manual annotations on depth maps for human pose estimation	Watch-R-Patch VGG- 19 CNN	Whole Body	21	Watch-n-Patch dataset	2019
10	Raaj et al. [41]	Efficient online multi-person 2D pose tracking with recurrent spatio-temporal affinity fields	RCNN	Whole body	21	COCO, MPII, and PoseTrack datasets	2019
11	Sharma et al. [42]	Monocular 3D human pose estimation by generation and ordinal ranking	Deep Conditional Variational Autoencoder	Whole body	16–17	CMU Motion Capture (Mocap), Human3.6 M, HumanEva-I	2019
12	Ershadi- Nasab et al. [43]	Multiple human 3D pose estimation from Multiview images	Full connected pairwise conditional random field	Whole body	14	Campus, Shelf, Utrecht Multi-Person Motion benchmark, Human3.6 M, KTH Football II, and MPII Cooking datasets	2018
13	Rogez et al. [44]	Image-Based Synthesis for Deep 3D Human Pose Estimation	CNN	Whole body	14	CMU motion capture dataset, Human3.6 M Dataset, Leeds Sports Dataset (LSP)	2018

Table 1 (continued)

S. No.	Authors	Title	Method/framework	HPE	Landmarks	Dataset	Year
14	Chang et al. [45]	V2V-PoseNet: Voxel-to-Voxel Prediction Network for Accurate 3D Hand and Human Pose Estimation from a Single Depth Map	3D point cloud-based encoder-decoder FCNN	Hand	14	Imperial Computer Vision and Learning Lab (ICVL) Hand Posture Dataset, NYU Hand Pose Dataset, MSRA Hand Pose Dataset, HANDS 2017 Frame-based 3D Hand Pose Estimation Challenge Dataset, ITOP Human Pose Dataset	2018
15	Cao et al. [46]	Realtime multi-person 2D pose estimation using part affinity fields	Two-branch multi-stage CNN	Whole body	14	COCO, MPII	2017
16	Wei et al. [47]	Convolutional pose machines	CNN	whole body	14	MPII, LSP, and Frames Labelled in Cinema (FLIC) datasets	2016
17	Tompson et al. [48]	Real-time continuous pose recovery of human hands using convolutional networks	CNN	Hand	14	RGBD sensor image dataset	2014
18	Shotton et al. [49]	Real-time human pose recognition in parts from single-depth images	Randomized decision forest	Whole body	31	Mocap dataset	2013

CMU Carnegie Mellon University, *AMASS* archive of motion capture as surface shape, *COCO* common objects in context, *NYU* New York University

Gao et al. [23] proposed a parallel deep neural network model to estimate the body pose and dual hand detection. In this work, ResNet-Inception layers and Single Shot MultiBox Detector (RI-SSD) are parallelly used to detect dual hands. On the other hand, VGG-19 architecture with the COCO dataset is used for human pose estimation. Based on the information on RI-SSD and VGG-19 architecture left and right hands are classified. The said information on hand detection with pose estimation is further tested on a second-generation astronaut assistant robot. Moreover, Hernandez et al. [24] presented a human pose estimation system using a double Kinect sensor to get the actual joint variables and locations. Kinect v2 and Albuquerque NM depth sensors are used to extract skeleton information of humans and this information is called ground truth. Further, two different state-of-art HPE frameworks namely OpenPose and Detectron 2 are used to compare the joint landmarks of human pos. Finally, the joint angles of the shoulder and elbow extracted from OpenPose and Detectron2 have been compared with ground *truth. McNally* et al. [12] proposed a neuro-evolution architecture that is based on a 2D convolution neural network along with a weight transfer function. The efficiency of the proposed model was increased using a multi-optimization method for validation loss. Jin et al. [25] developed a top-down approach called ZoomNet which is based on Faster RCNN and a new COCO-whole body dataset with manual annotation of four bounding boxes and 133 key points. Tu et al. [26] discussed a cuboidal proposal network (CPN) with a pose regression network (PRN) which is based on voxel-to-voxel network 3D convolutions as a building block. Dai et al. [11] proposed a cascaded hierarchical CNN architecture known as 4CHNet for RGB image-based 3D hand pose estimation. Plantard et al. [27] presented the Kinect sensor-based ergonomic analysis of virtual mannequin posture analysis. In this work, the joint landmarks along with rapid upper body assessment (RULA) analysis have been made using a Kinect sensor. Bashirov et al. [28] developed real-time RGB depth-based pose estimation in 3D. For obtaining the real-time pose estimation, hand pose, and facial expression the authors used Kinect RGB-D camera. In addition, Zhang et al. [29] proposed a new method for pose estimation using a Kinect sensor with a perspective n-points (PnP) algorithm. The proposed PnP algorithm is used to get the relative position of various cameras and to map real 3D points of space with the 2D camera image. Sarsfield et al. [30] introduced a clinical assessment of human posture using a Kinect sensor. The authors performed a comprehensive analysis for pose estimation in rehabilitation applications. They worked on upper body pose estimation for stroke rehabilitation cases. They concluded that pose estimation yields significant errors when comparing the joint variables of the shoulder, arm, and elbow. Saeed et al. [5] proposed a frame-based approach for head pose estimation using a haar-cascade algorithm. They created a frame using a 2D color image with a 3D depth point cloud using feature extraction. Wu et al. [31] discussed a model based recursive matching algorithm for the pose estimation. This algorithm uses a 2D image with 3D point cloud data as an input for further training the model to fit. The proposed algorithm has been compared with Kinect real-time pose estimation and the obtained results shows higher accuracy. Obdrzalek et al. [32] presented the accuracy of joint localization and robustness of pose estimation with respect to orientation and occlusion using a Kinect sensor. They have used an impulse motion capture system for tracking LED markers attached to various joint locations. This work gives the accuracy of Kinect pose estimation using motion capture for the training of elderly people. Further, a more detailed and comprehensive study of the works of literature can be found in an article by Bazarevsky et al. [33].

Based on the above literature, it has been observed that many researchers are contributing to deep learning-based pose-tracking algorithms. On the other hand, Kinect v1 and v2 sensors are frequently used to create 3D point clouds and datasets for further HPE. The main challenge of the HPE algorithm is real-time implementation and minimization of joint angle errors. Therefore, a real-time inverse kinematic solver is employed to calculate the joint angles for the given elbow-wrist coordinates. These methods are quite accurate and have also been implemented on various robots. On the other hand, OpenPose, HMR, OpenNI, VoxelNet, PoseNet, etc. as discussed in the works of literature are the most popular pose estimation algorithms and are being adopted by many researchers. Apart from these state-of-the-art algorithms, the MediaPipe framework also yields minimum error and perfectly classifies the various joint landmarks. As per the authors' knowledge, the real-time positioning of a humanoid robot arm using the MediaPipe framework is not reported. Also, the performance of the MediaPipe pose estimation framework in terms of the standard error is missing. The current research article mainly deals with the Kinect sensor-based skeleton tracking and MediaPipe HPE framework for the extraction of joint angles of human pose landmarks and its implementation on real-time humanoid robot prototypes. The performance of the adopted algorithms has been compared in terms of standard error. The main contributions of this work include a comprehensive study of various HPE algorithms and their implementation in real-time. The authors also developed a 3D-printed robot prototype used to implement the HPE framework. Also, two different methods Kinect-based skeleton tracking [49] and MediaPipe [33, 50, 51] frameworks are considered for pose estimation. Later on, an inverse kinematic algorithm is used to calculate the joint angles of a real-time robot as well as the adopted HPE framework. Comparison has been made in terms of joint angles for the adopted framework and also with a real-time robot. Finally, the standard error for all joint landmarks and arm angles is calculated. It was found that the standard error for the MediaPipe-based solution was less as compared to Kinect based skeleton tracking method. Jong et al. [52] discussed the combination of a more sophisticated humanoid model and a fast optimization method to estimate the joint angles of 3D pose estimation based on a humanoid model. Further, Alberto et al. [53] proposed a systematic procedure for collaborative tasks in a dynamic environment. The proposed methodology mainly focuses on the contribution and the mapping of reference frames.

2 Mathematical formulation and its algorithms

Many researchers have developed multiple pose estimation algorithms but, these algorithms can be based on learning approaches or human model-based approaches. These methods act as the building block for joint tracking and pose estimations. The most conventional approach is to calculate the joint angles using inverse kinematic (IK) algorithms which yield fast and accurate results based on the given end effector position and orientation. To test the IK algorithm along with HPE frameworks, a custom 3D-printed humanoid robot prototype is used which is shown in Fig. 1. The prototype humanoid robot is equipped with micro servo motors in all joints.

2.1 Forward and inverse kinematics

The kinematics of the humanoid robot's upper arm is solved by using an analytical approach. It consists of both forward and inverse kinematic equations. Initially, the forward kinematics of the robotic manipulator is solved after assigning the coordinate frames at each joint of the humanoid robotic arm to obtain the Position and orientation of the end effector. Figure 2 shows the assigning of the coordinate frames at each joint of the robotic arm. Once the forward kinematics approach is solved based on the position and orientation of the end effector the authors used the inverse kinematics approach for obtaining the joint angles. The mathematical equations related to inverse kinematics are mentioned in Eqs. (1) and (2).

$$\theta_2 = \pm cos^{-1}\left(\frac{\|X\|^2 - l_1^2 - l_2^2}{2l_1l_2}\right) \tag{1}$$

$$\theta_1^i = \theta - \theta^i \tag{2}$$

where $\theta = atan2(Y1, X1)$, $\theta^i = atan2(l2sin\theta_2, l1 + l2cos\theta_2)$.

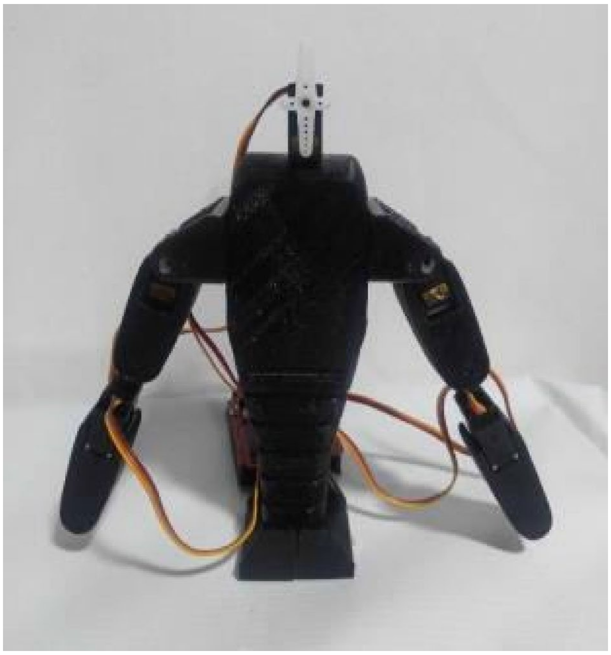


Fig. 1 Prototype 3D printed humanoid robot

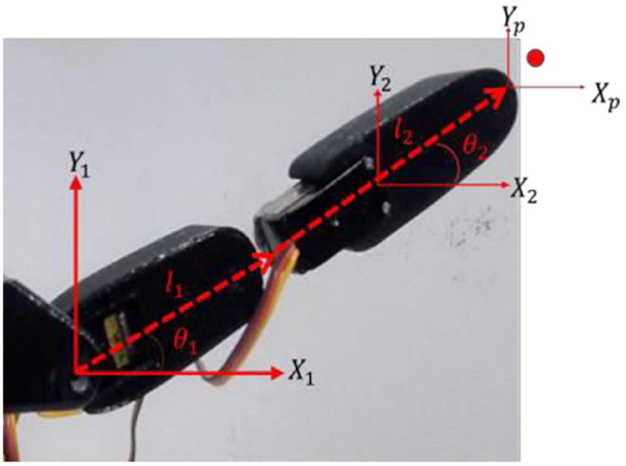


Fig. 2 Coordinate frames assigned at each joint of the robotic arm

The algorithm for the upper arm is given as follows:
IK Algorithm

```
1.  $cos\theta_2 = \frac{\|X\|^2 - l_1^2 - l_2^2}{2l_1l_2}$
2.  $If\, cos\theta_2 > 1$then return $\emptyset$
3.  $If\, cos\theta_2 = 1$then return $\{atan2(Y_1, X_{1)}, 0\}$
4.  $If\, cos\theta_2 = -1 and X \neq 0$ then return $\{atan2(Y_1, X_{1)}, \pi\}$
5.  $If\, cos\theta_2 = -1 and X = 0$ then return $\{\theta_1 \pi\} where \theta_1 \in (0, \pi)$
6.  $Else\ \theta_2^i = \pm cos_2^{-i}$ where i=1,2
7.  Calculate $\theta = atan2(Y_1, X_{1)}$
8.  $for\ i = (1to2)do$
        $\theta_1^i = \theta - \theta^i$
    $return(\theta_1^i and \theta_2^i)$
```

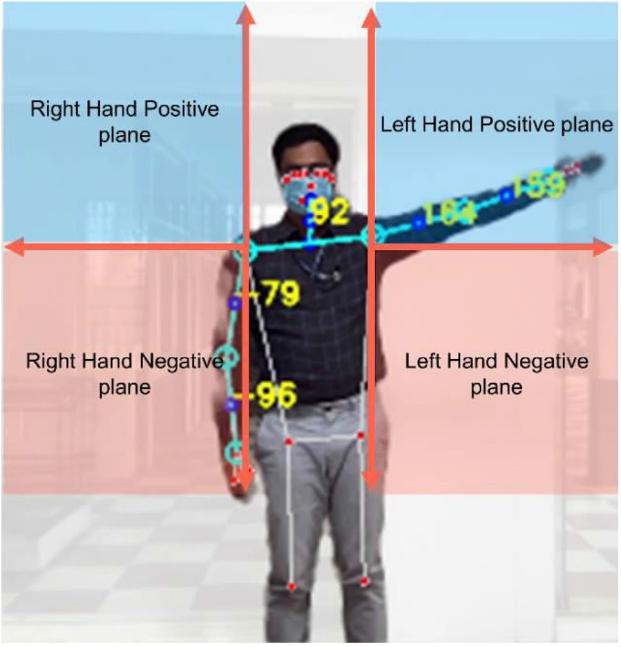
2.2 MediaPipe based HPE

The earlier discussed Inverse kinematic algorithm can be further implemented on the MediaPipe HPE framework to calculate the joint angles and position of the human arm. These joint angles are configured by positive and negative planes as shown in Fig. 3. If the hand falls in a positive plane the joint angle is calculated based on the arc tangent of the wrist coordinate while negative angles are calculated when the hand falls on a negative plane. Based on this concept, the position control of the robotic arm and its joint variables are communicated through the python-Arduino pyserial library. These obtained joint variables are communicated every millisecond and based on the received joint information, the robot arm mimics human gestures. The detected joint landmarks and corresponding joint angles are calculated using an inverse kinematic algorithm.

Further, the proposed MediaPipe graph for the pose estimation is shown in Fig. 4a, b. The proposed flow chart shows the flow and node connectivity of the proposed framework. The flow chart requires the input that is, audio or video which can be proceeded or transformed by its modular components shown in yellow and light blue components. These components are also known as a pipeline. Each pipeline is connected to specific input and output nodes and these nodes in the flow chart are implemented as a calculator. Figures 5 and 6 consists of PoseTracking and PoseRenderer components. Moreover, MediaPipe consists of three major components:

1. Input framework for sensory information (i.e., audio/video),
2. Tools for performance evaluations, and

Fig. 3 Coordinate planes for positive and negative joint angles



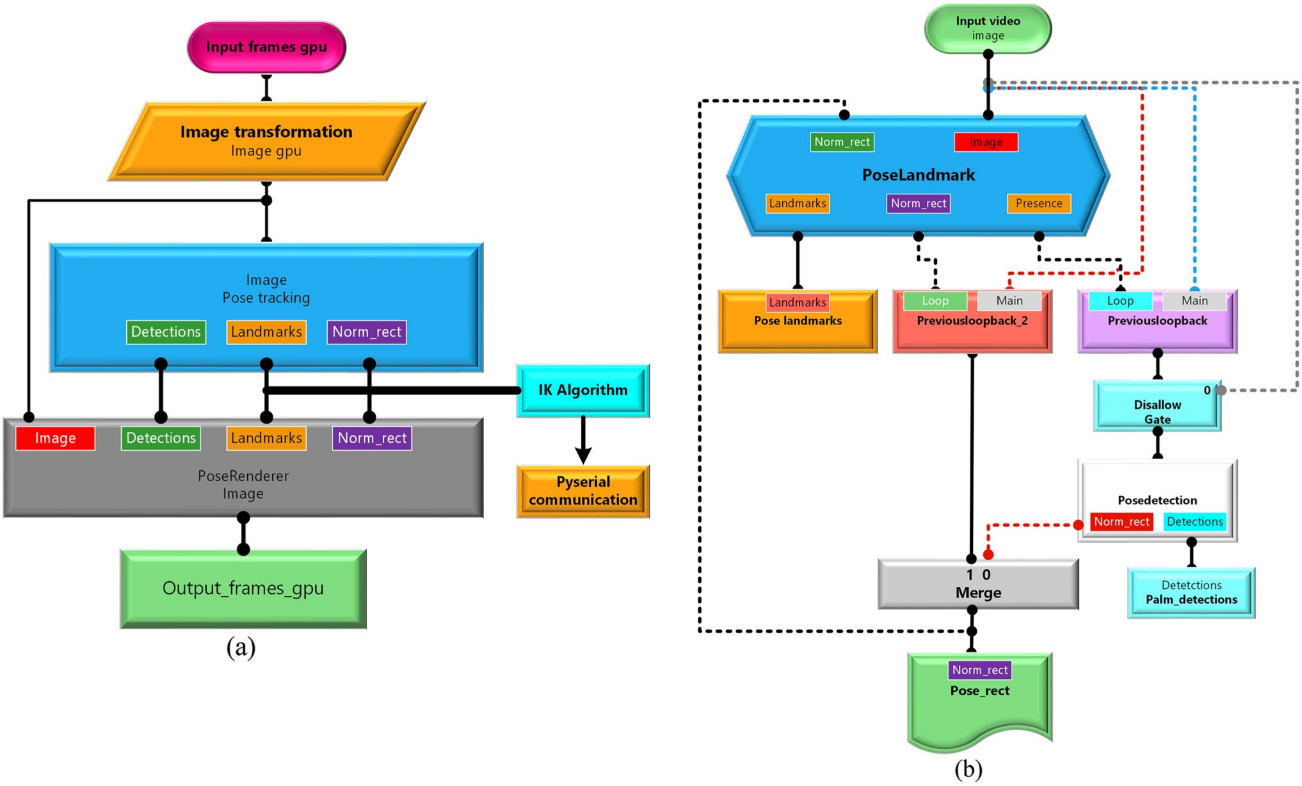
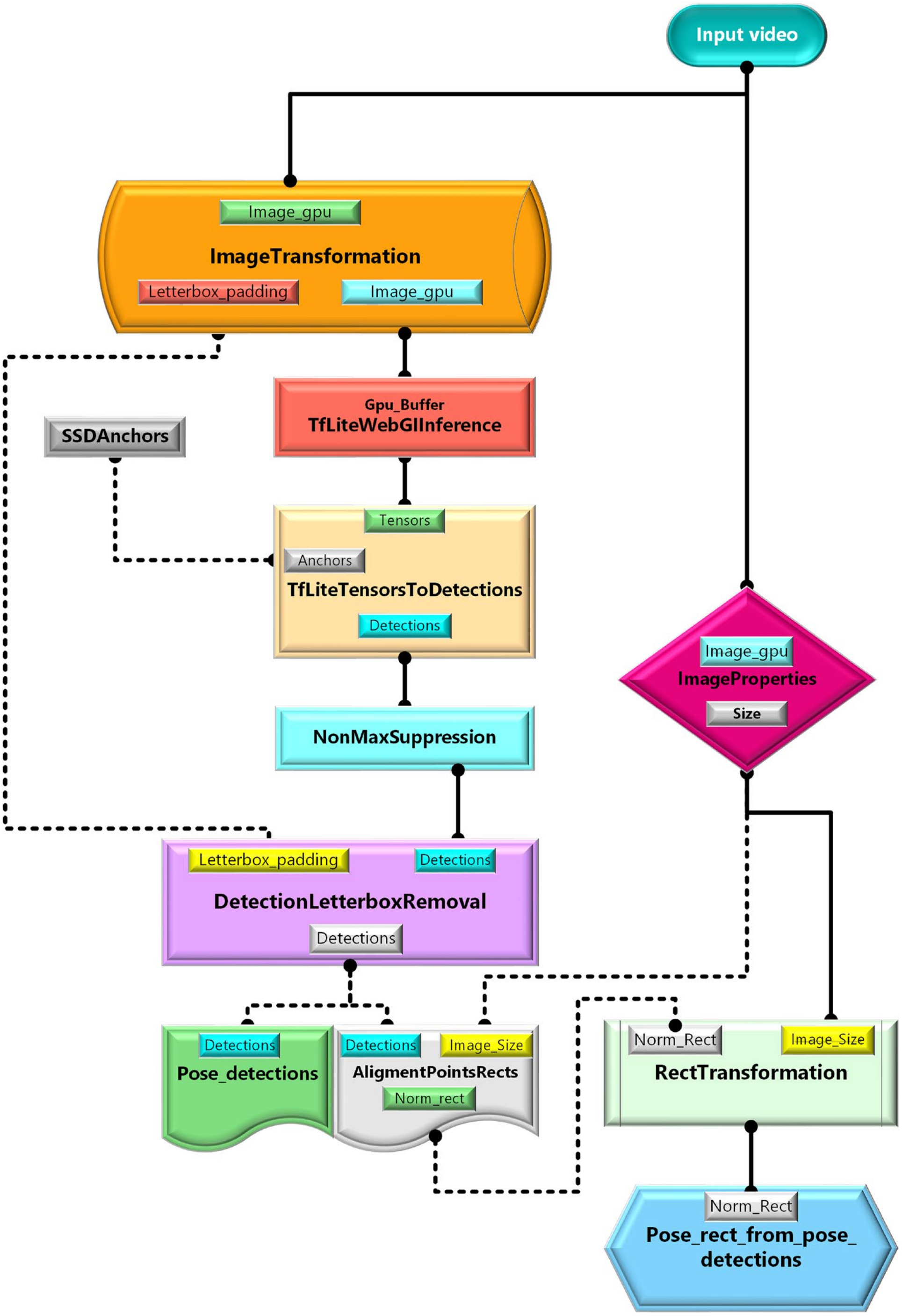


Fig. 4 Flow chart shows **a** MediaPipe main pipeline **b** pose tracking procedure

3. Processing components known as calculators.

These components are the backbone for pose detection, object tracking, image segmentation, motion tracking, box tracing, etc. However, there are many HPE models have been proposed in recent years but MediaPipe is one of the most efficient frameworks developed to build various machine learning-based solutions. It has the flexibility to deploy mobile,

Fig. 5 Flow chart shows the pose detection procedure



web, edge, or cloud-based applications. Therefore, leveraging this framework for controlling the humanoid robot arm in real-time.

In Fig. 4a, "Input_frames_gpu" specifies the input to the graph which contains default 100 frames that can be queued for further processing. This node is further connected with the "Image Transformation" calculator which flips the input image horizontally. Node "Pose Tracking" This node performs pose tracking using a subgraph calculator: "Pose Tracking Subgraph" uses the flipped input_frames_gpu as input image and pose landmarks outputs normalized rectangle information with pose detections. At last, the Pose Renderer Subgraph Node renders the pose on the input frames using the "PoseRendererSubgraph" calculator. Multiple input streams: Takes the flipped images, pose landmarks, normalized rectangle, and pose detections as input and outputs the final frames with rendered poses to the "output_frames_gpu" stream.

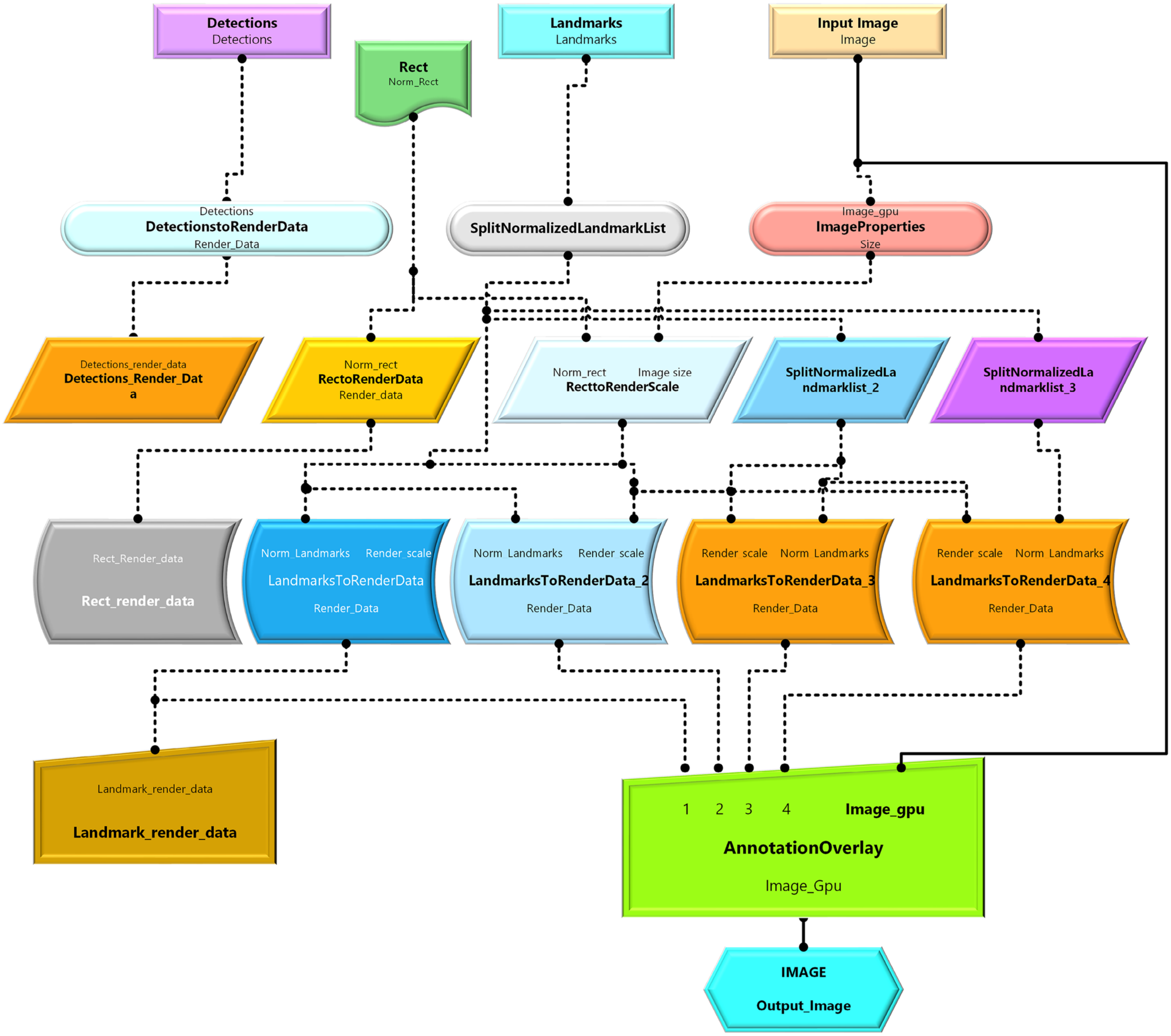


Fig. 6 Flowchart shows the pose renderer procedure

In summary, this MediaPipe graph takes input frames from "input_frames_gpu," performs image transformation, flips the frames horizontally, then uses a subgraph for pose tracking, and finally renders the poses on the input frames, producing the output frames in the "output_frames_gpu" stream. Figure 4b depicts the subgraph within the MediaPipe framework, specifically for pose tracking. This subgraph is referenced in the main graph as a node with the type "PoseTrackingSubgraph." The subgraph takes an input video stream, performs various processing steps related to pose detection and landmark localization, and outputs pose-related information. In summary, this subgraph processes input video frames, performs pose detection and landmark localization, and outputs pose-related information, such as landmarks, normalized rectangles, and pose detections. The flow is controlled based on the presence of a pose in the previous frame, and feedback mechanisms ensure continuity in decision-making across frames.

Figures 5 and 6 consists of PoseTracking and PoseRenderer components. MediaPipe subgraph specifically for pose detection. This subgraph is used in the larger pipeline described in the previous responses. In summary, this subgraph takes an input video, transforms the images, runs a pose detection model, performs post-processing steps such as non-max suppression, adjusts detections for letterboxing, and outputs the final pose-related information, including pose detections and normalized rectangles. This subgraph is part of the overall pose-tracking pipeline described in the previous responses. Figure 6 depicts another subgraph in the MediaPipe framework, specifically for rendering the results of the pose tracking pipeline. This subgraph is referenced in the main graph as a node with the type "PoseRendererSubgraph". In summary, this subgraph takes input streams containing pose detections, landmarks, and normalized rectangles, calculates the necessary rendering information, and outputs a final rendered image with annotations and overlays. The rendered image is then used as part of the overall pose-tracking pipeline described in the previous responses.

2.3 RGB-D-based skeleton tracking

The main challenge of these computer vision-based algorithms is to calculate the depth in real-time. This depth of information is crucial to avoid the uncertainty present in the environment and also to grasp any object of concern. To face these challenges, the Microsoft Kinect sensor can be used to calculate the depth information and position control of the robot arm. On the other hand, the mapping of multiple joint coordinate frames with respect to human pose landmarks is quite noisy and inaccurate for RGB cameras. Even though the reference coordinate of joint landmarks varies with respect to each frame the Kinect infrared (IR) sensor provides exact information. IR sensors with RGB sensors create 3D point clouds or depth profiles of an object. These points can be further used to create the joint landmarks of the human pose [51–55]. The wrist coordinates are extracted from the skeleton and fed to the inverse kinematics solver which is helpful to calculate the joint angles. These joint angles are communicated through a pyserial module. In the current research work, the authors considered Python 3.7.5 and pyserial 3.5 versions. Figure 7 shows the basic steps of the Kinect-based HPE approach [56].

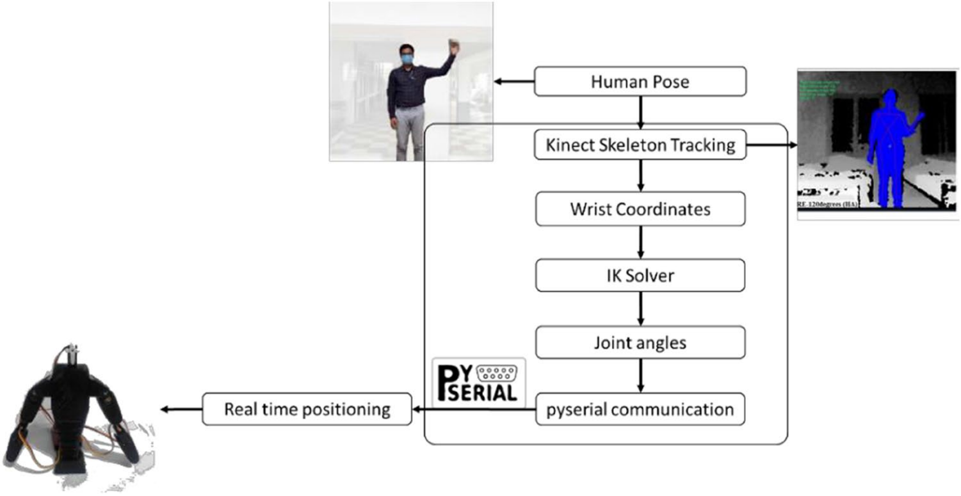
3 Results and discussion

The performance of each framework discussed in the previous section is analyzed individually. Further, the evaluation of accuracies in terms of analytical inverse kinematic solutions is compared. Later on, real-time joint angles are recorded for calculating the error. A comparison has been made in terms of the standard error of Kinect-based skeleton tracking and MediaPipe framework as shown in Fig. 8.

Figure 9a–e shows the various joint angles such as left and right shoulder elbow and elbow wrist, and head angles obtained from the robot, Mediapipe, and Kinect sensor. It has been observed that the joint angles obtained from the Kinect sensor cause multiple misclassifications when compared to MediaPipe. In Mediapipe the joint angle data is obtained from the normal webcam which produces a better result than the Kinect sensor.

Further, the proposed system shows the robustness of the adopted pose estimation framework by using HPE. The details of various joint angles produced by all frameworks are collected and saved in a csv file and a sample of the collected data is represented in a boxplot as shown in Fig. 10. It has been observed that all the dynamic poses are perfectly mimicked and mapped onto real-time humanoid robot arm control. Moreover, a sample of frames recorded from the Kinect-based skeleton tracking is shown in Fig. 11. Figure 11 shows the 18 different dynamic poses depicted by the Kinect sensor; all these samples are collected from recorded video.

Fig. 7 Flow chart shows the overall structure of the proposed framework



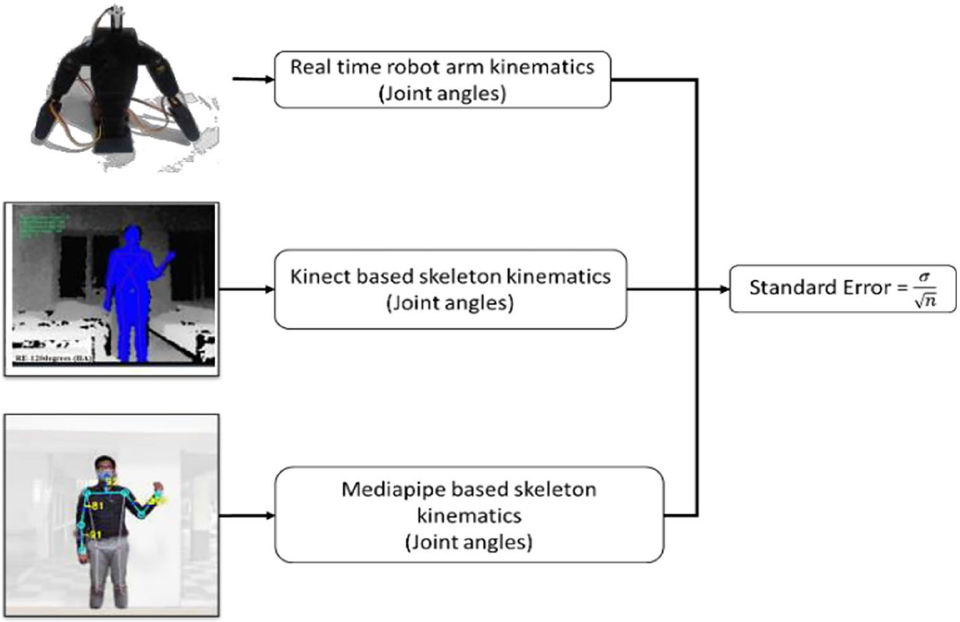

Real time robot arm kinematics
(Joint angles)
Kinect based skeleton kinematics
(Joint angles)
Mediapipe based skeleton
kinematics
(Joint angles)
Standard Error = $\frac{\sigma}{\sqrt{n}}$

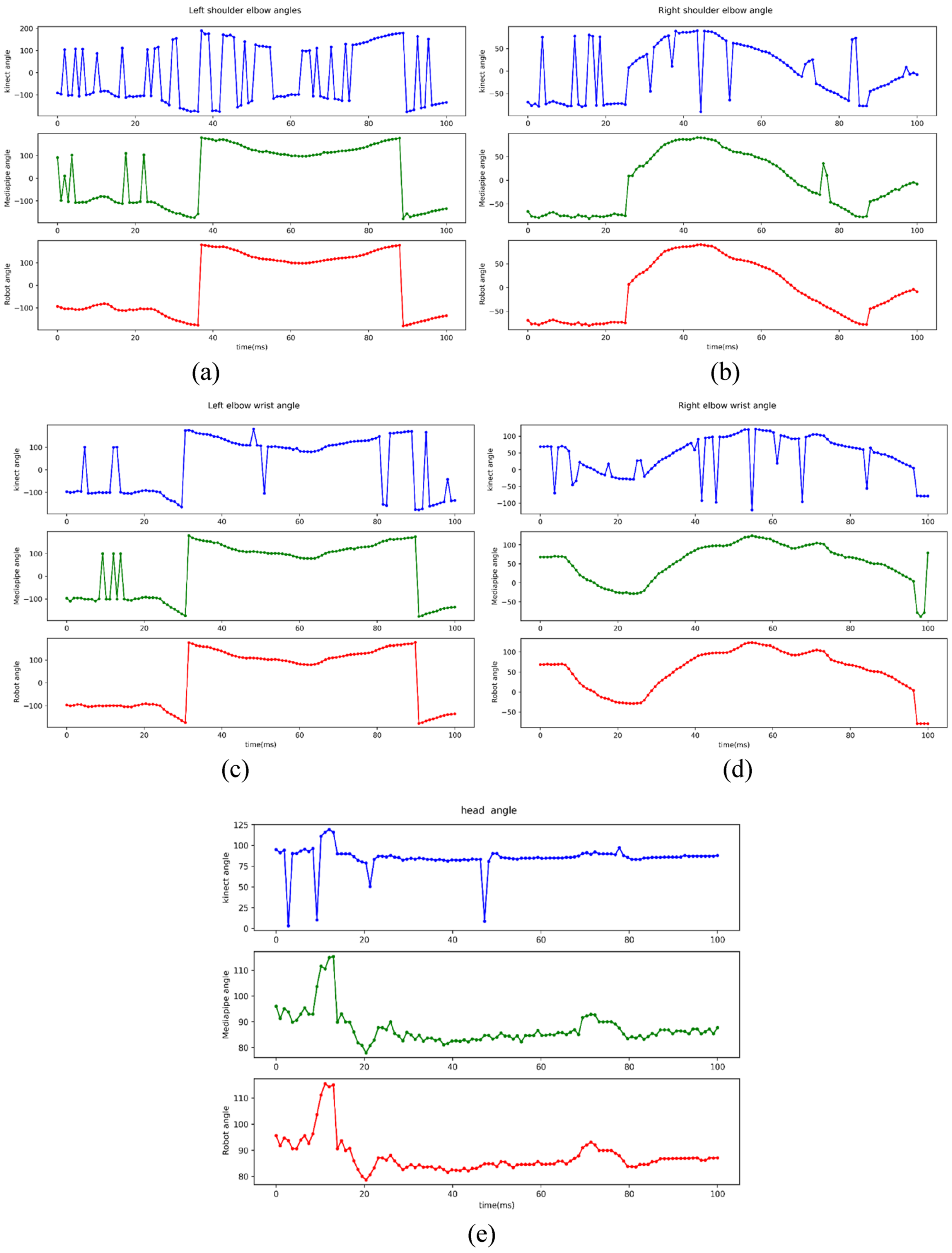
Fig. 8 Standard error-based comparison of all joint variables

Figure 12 shows the different postures and corresponding angles obtained from the MediaPipe framework. Similar to the Kinect sensor, here also obtained eighteen different dynamic poses. Further, Fig. 13 shows that few samples are obtained in real-time 3D world coordinates of human postures. These 3D coordinates are measured in meters with the origin at the hip center. Based on the concept of positive and negative planes, the angles are shown accordingly. Finally, real-time dynamic control of the humanoid robotic arm using these frameworks is shown in Fig. 14. Although it is quite difficult to analyze these pictorial representations of pose and corresponding joint angles. Therefore, standard errors are discussed in the next section.

Figure 15 shows the error bar plot for each key point of a human pose as well as humanoid robot joint angles. These plots were drawn by collecting the postures data in real time. In the current research work, the authors used the Python matplotlib library is leveraged for plotting all the graphs. It is visible from the error bar plot that Kinect-based joint angles are far from the real-time robot joint angles. As already discussed, the number of outliers is also present in Kinect-based positioning. These error bars are coded with orange, red, and green colors for better visualizations. The length of the cap or capsize gives the error between the actual and predicted angles from all frameworks. Furthermore, the standard errors produced by each framework are given in Table 2. It has been observed that the error produced by Kinect-based joint angles compared to MediaPipe is maximum. The standard error for REW is 3.72 and for head angles, it is 0.7 as compared to MediaPipe.

4 Conclusions

A human pose estimation framework based on real-time position control of a humanoid robot arm has been presented in this work. Initially, the proposed human and robot joint angles are captured from RGBD and 2D video webcams in real-time. Later on, the said proposed system is captured from Kinect-based skeleton tracking and the MediaPipe framework. Based on the obtained results, the position control of the humanoid robot arm using the MediaPipe pose framework with a regular webcam is also feasible. Although depth-based estimations are more popular, the availability of such platforms is not as common as compared to regular webcams or USB cameras. It is evident from the results that the MediaPipe framework tends to outperform when compared to Kinect-based skeleton tracking in all possible joint movements. The result shows that the robot can mimic a human pose in real time, regardless of surrounding luminescence or the presence of an unknown user in the frame. Moreover, the results of the comparison are demonstrated for both static and dynamic conditions of human body movement. Further, more complex robot configurations can also be considered for the development of human–robot interactions. The proposed frameworks are efficient and produce less error. Therefore, these frameworks can also be implemented for gesture-based control, medical rehabilitation, and assisting the elderly.



(a)

(b)

(c)

(d)

(e)

Fig. 9 Various joint angles obtained from MediaPipe, and the Kinect sensor of the robot. **a** Left shoulder elbow, **b** right shoulder elbow, **c** left elbow wrist, **d** right elbow wrist, and **e** head

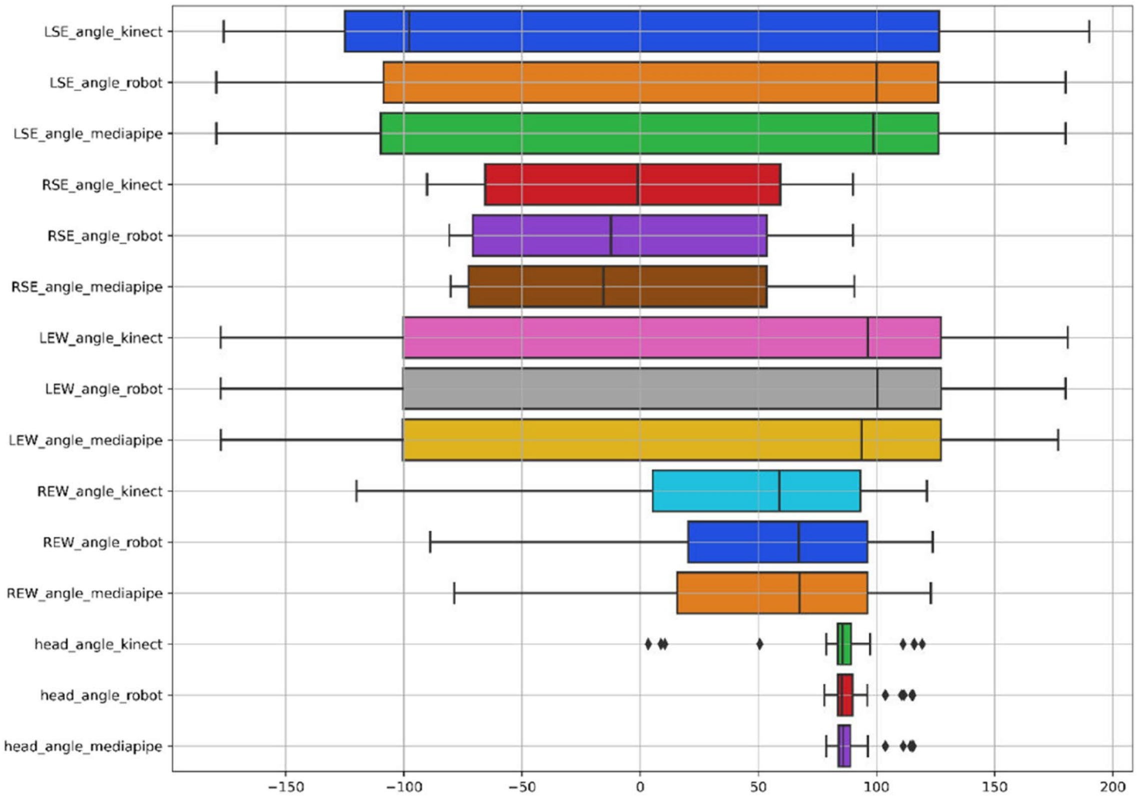


Fig. 10 Various joint angles are shown in the boxplot

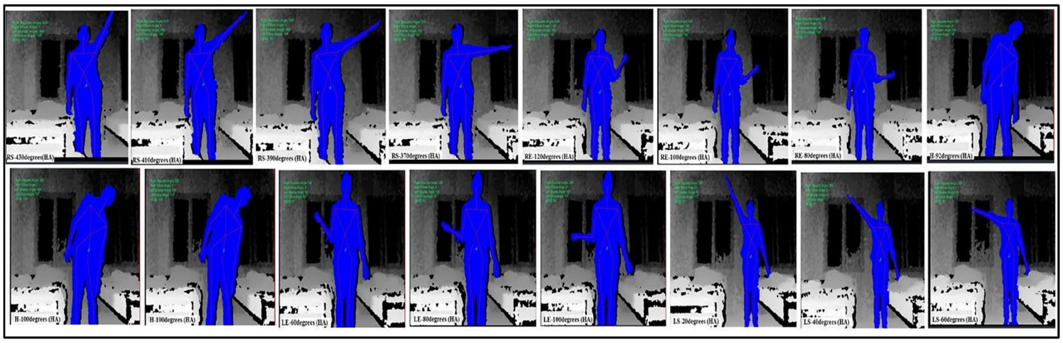


Fig. 11 Kinect sensor-based skeleton tracking and corresponding joint angles

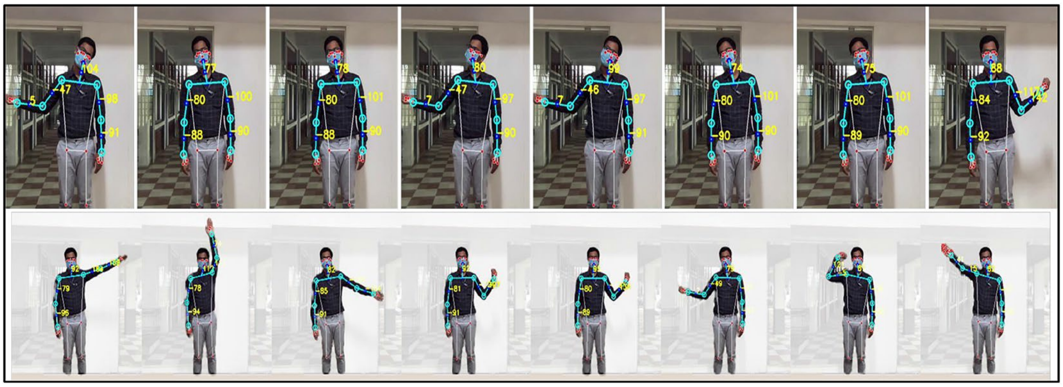


Fig. 12 MediaPipe-based joint landmarks and angles

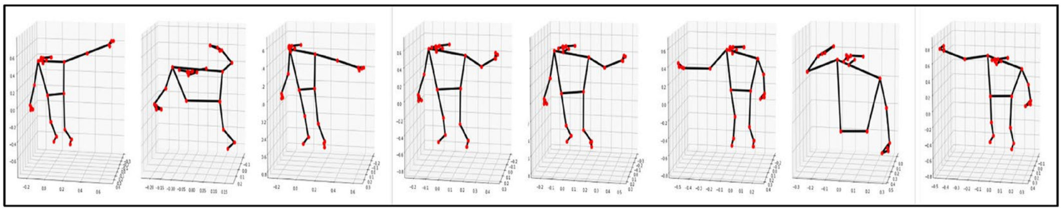


Fig. 13 Samples of MediaPipe pose estimation real-world 3D coordinates

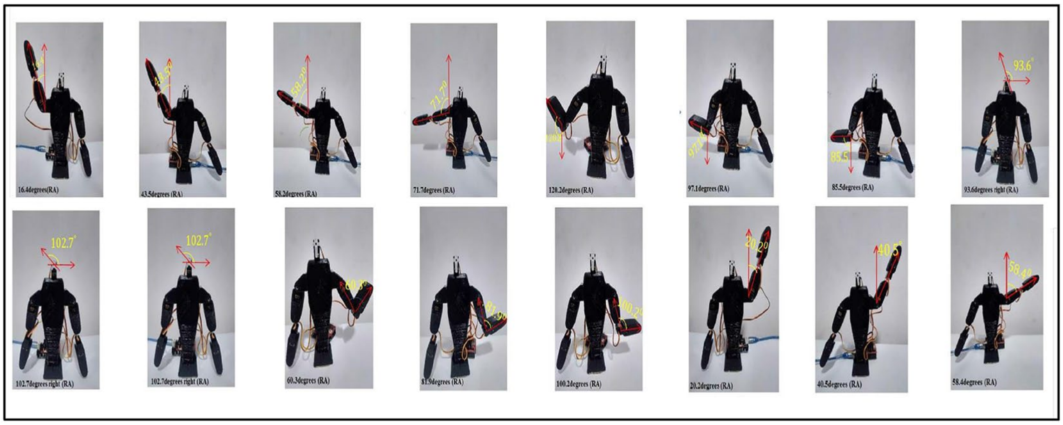


Fig. 14 Real-time joint angles and positioning of the robotic hand

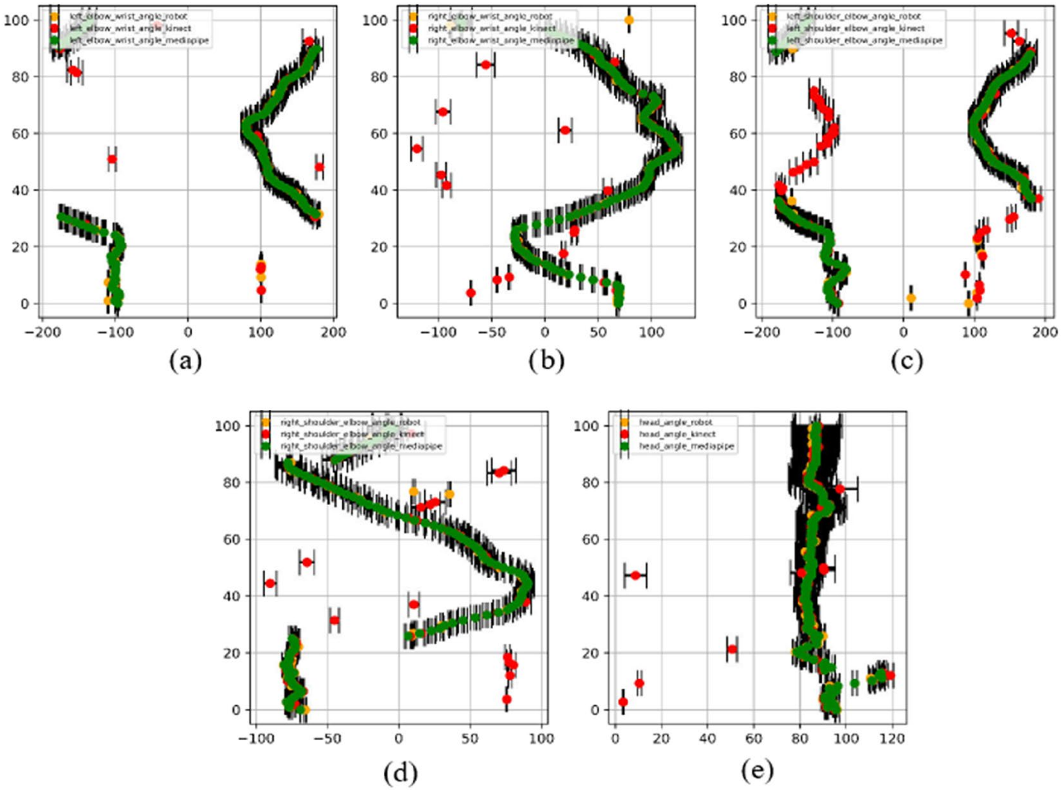


Fig. 15 Error bar plot various joint angles **a** LEW, **b** REW, **c** LSE, **d** RSE, and **e** Head

Table 2 Standard error comparison between Robot, Kinect, and MediaPipe joint angles

Joint angle landmarks	Standard error between robot and Kinect joint angles	Standard error between robot and MediaPipe joint angles
LSE angles	9.05954253	9.005756
RSE angles	4.08329804	4.083841
LEW angles	8.36706216	8.384548
REW angles	3.72791752	3.417926
head angles	0.7861822	0.43512

LSE left shoulder elbow, *RSE* right shoulder elbow, *LEW* left elbow wrist, *REW* right elbow wrist

Author contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Nuthalapati Hemalatha, Mehmet Şükrü Adin, and Ravi Kumar Mandava. The first draft of the manuscript was written by Panchanand Jha and G. Praveen Kumar Yadav. Final draft was prepared and evaluated by Din Bandhu, Kuldeep K Saxena and Mahaboob Patel. All authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Funding The authors have not disclosed any funding.

Data availability All data generated or analyzed during this study are included in this published article.

Declarations

Competing interests The authors declare that they have no conflict of interest.

Ethics approval and consent to participate All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee and with the 1964 Helsinki Declaration and its later amendments or comparable ethical standards. Approval was obtained from the ethics committee of Raghu Engineering College, Visakhapatnam, Andhra Pradesh, India.

Informed consent All the authors were involved in the study and all of them have provided their consent for image capturing to study human–machine interaction.

References

1. Kahraman C, Deveci M, Boltürk E, et al. Fuzzy controlled humanoid robots: a literature review. Rob Auton Syst. 2020;134:103643.
2. Gorade U, Bandhu D, Kumari S, et al. Design of bluetooth-controlled floor cleaning robot. In: *Recent advances in mechanical infrastructure*. Springer, Singapore, pp. 121–131
3. Moeslund TB, Hilton A, Krüger V. A survey of advances in vision-based human motion capture and analysis. Comput Vis Image Underst. 2006;104:90–126.
4. Dang Q, Yin J, Wang B, et al. Deep learning based 2D human pose estimation: a survey. Tsinghua Sci Technol. 2019;24:663–76.
5. Saeed A, Al-Hamadi A, Ghoneim A. Head pose estimation on top of haar-like face detection: a study using the Kinect sensor. Sensors (Switz). 2015;15:20945–66.
6. Spehr J. Human pose estimation. In: On hierarchical models for visual recognition and learning of objects, scenes, and activities. Studies in systems, decision and control, vol 11. Springer, Cham. https://doi.org/10.1007/978-3-319-11325-8_6
7. Toshpulatov M, Lee W, Lee S, et al. Human pose, hand and mesh estimation using deep learning: a survey. J Supercomput. 2022;78:7616–54.
8. Mwiti DA. Guide to human pose estimation | by Derrick Mwiti | heartbeat. Hear Comet. 2019;2019:1–7.
9. Wu J, Trivedi MM. A two-stage head pose estimation framework and evaluation. Pattern Recognit. 2008;41:1138–58.
10. Osokin D. Real-time 2D multi-person pose estimation on CPU: Lightweight OpenPose. In *ICPRAM 2019—proceedings of the 8th international conference on pattern recognition applications and methods*. 2019; pp. 744–748
11. Dai S, Liu W, Yang W, et al. Cascaded hierarchical CNN for RGB-based 3D hand pose estimation. Math Probl Eng. 2020;2020:1–13.

12. McNally W, Vats K, Wong A, et al. EvoPose2D: pushing the boundaries of 2D human pose estimation using accelerated neuroevolution with weight transfer. IEEE Access. 2021;9:139403–14.
13. Cantarini G, Tomenotti FF, Noceti N, et al. HHP-Net: a light Heteroscedastic neural network for Head Pose estimation with uncertainty. In: *Proceedings—2022 IEEE/CVF winter conference on applications of computer vision, WACV 2022*. 2022; pp. 3341–3350
14. Madrigal F, Lerasle F. Robust head pose estimation based on key frames for human-machine interaction. Eurasip J Image Video Process. 2020;2020:1–19.
15. Martin JB, Moutarde F. Real-time gestural control of robot manipulator through deep learning human-pose inference. In: *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)*, pp. 565–572
16. Chamorro S, Collier J, Grondin F. Neural network based lidar gesture recognition for realtime robot teleoperation. In: 2021 IEEE international symposium on safety, security, and rescue robotics, SSRR 2021. Institute of Electrical and Electronics Engineers Inc., 2021; pp. 98–103
17. Zimmermann C, Welschehold T, Dornhege C, et al. 3D Human pose estimation in RGBD images for robotic task learning. In: Proceedings—IEEE international conference on robotics and automation. Institute of Electrical and Electronics Engineers Inc., pp. 1986–1992
18. Gago JJ, Vasco V, Łukawski B, et al. Sequence-to-sequence natural language to humanoid robot sign language. Epub ahead of print 9 July 2019. https://doi.org/10.11128/arep.58
19. Amini A, Rosman G, Karaman S, et al. Variational end-to-end navigation and localization. In: Proceedings—IEEE international conference on robotics and automation. Institute of Electrical and Electronics Engineers Inc., 2019; pp. 8958–8964
20. Michel D, Qammaz A, Argyros AA. Markerless 3D human pose estimation and tracking based on RGBD cameras: an experimental evaluation. In: ACM international conference proceeding series, pp. 115–122
21. Liang CJ, Lundeen KM, McGee W, et al. A vision-based marker-less pose estimation system for articulated construction robots. Autom Constr. 2019;104:80–94.
22. Cai L, Ma Y, Xiong S, et al. Validity and reliability of upper limb functional assessment using the microsoft kinect V2 sensor. Appl Bionics Biomech. 2019;2019:01–14.
23. Gao Q, Liu J, Ju Z, et al. Dual-hand detection for human-robot interaction by a parallel network based on hand detection and body pose estimation. IEEE Trans Ind Electron. 2019;66:9663–72.
24. Hernández ÓG, Morell V, Ramon JL, et al. Human pose detection for robotic-assisted and rehabilitation environments. Appl Sci. 2021;11:4183.
25. Jin S, Xu L, Xu J, et al. Whole-body human pose estimation in the wild. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer Science and Business Media Deutschland GmbH, pp. 196–214
26. Tu H, Wang C, Zeng W. VoxelPose: towards multi-camera 3D human pose estimation in wild environment. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer Science and Business Media Deutschland GmbH, pp. 197–212
27. Plantard P, Auvinet E, Le Pierres AS, et al. Pose estimation with a kinect for ergonomic studies: evaluation of the accuracy using a virtual mannequin. Sensors (Switz). 2015;15:1785–803.
28. Bashirov R, Ianina A, Iskakov K, et al. Real-time RGBD-based extended body pose estimation. In: Proceedings—2021 IEEE winter conference on applications of computer vision, WACV 2021. Institute of Electrical and Electronics Engineers Inc., pp. 2806–2815
29. Zhang S, Yu H, Dong J, et al. Combining kinect and PnP for camera pose estimation. In: Proceedings—2015 8th international conference on human system interaction, HSI 2015. Institute of Electrical and Electronics Engineers Inc., 2015; pp. 357–361
30. Sarsfield J, Brown D, Sherkat N, et al. Clinical assessment of depth sensor based pose estimation algorithms for technology supervised rehabilitation applications. Int J Med Inform. 2019;121:30–8.
31. Wu Q, Xu G, Li M, et al. Human pose estimation method based on single depth image. IET Comput Vis. 2018;12:919–24.
32. Obdrzalek S, Kurillo G, Ofli F, et al. Accuracy and robustness of Kinect pose estimation in the context of coaching of elderly population. In: Proceedings of the annual international conference of the IEEE engineering in medicine and biology society, EMBS. 2012; pp. 1188–1193
33. Bazarevsky V, Grishchenko I, Raveendran K, et al. BlazePose: on-device real-time body pose tracking. In: CVPR workshop on computer vision for augmented and virtual reality, Seattle, WA, USA, 2020; pp. 1–4
34. Wan TJ, Luo YM, Zhang Z, et al. TSNet: tree structure network for human pose estimation. Signal Image Video Process. 2022;16:551–8.
35. Zhang Z, Wang C, Qiu W, et al. AdaFuse: adaptive multiview fusion for accurate human pose estimation in the wild. Int J Comput Vis. 2021;129:703–18.
36. D'Eusanio A, Pini S, Borghi G, et al. RefiNet: 3D Human pose refinement with depth maps. In: Proceedings—international conference on pattern recognition. Institute of Electrical and Electronics Engineers Inc., 2020; pp. 2320–2327
37. Miki D, Abe S, Chen S, et al. Robust human pose estimation from distorted wide-angle images through iterative search of transformation parameters. Signal Image Video Process. 2020;14:693–700.
38. Miura T, Sako S. 3D human pose estimation model using location-maps for distorted and disconnected images by a wearable omnidirectional camera. IPSJ Trans Comput Vis Appl. 2020;12:1–17.
39. Chen Y, Ma H, Kong D, et al. Nonparametric structure regularization machine for 2D hand pose estimation. In: Proceedings—2020 IEEE winter conference on applications of computer vision, WACV 2020. Institute of Electrical and Electronics Engineers Inc., pp. 370–379
40. D'Eusanio A, Pini S, Borghi G, et al. Manual annotations on depth maps for human pose estimation. In: Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics). Springer, pp. 233–244
41. Raaj Y, Idrees H, Hidalgo G, et al. Efficient online multi-person 2D pose tracking with recurrent spatio-temporal affinity fields. In: Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 2018; 2019-June: 4615–4623
42. Sharma S, Varigonda PT, Bindal P, et al. Monocular 3D human pose estimation by generation and ordinal ranking. In: Proc IEEE Int Conf Comput Vis 2019; 2019-October: 2325–2334
43. Ershadi-Nasab S, Noury E, Kasaei S, et al. Multiple human 3D pose estimation from multiview images. Multimed Tools Appl. 2018;77:15573–601.
44. Rogez G, Schmid C. Image-based synthesis for deep 3D human pose estimation. Int J Comput Vis. 2018;126:993–1008.

45. Chang JY, Moon G, Lee KM. V2V-PoseNet: voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map. In: Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 2017; 5079–5088
46. Cao Z, Simon T, Wei SE, et al. Realtime multi-person 2D pose estimation using part affinity fields. In: Proc—30th IEEE Conf Comput Vis Pattern Recognition, CVPR 2017 2016; 2017-January: 1302–1310
47. Wei SE, Ramakrishna V, Kanade T, et al. Convolutional pose machines. In: Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 2016; 2016-December: 4724–4732
48. Tompson J, Stein M, Lecun Y, et al. Real-time continuous pose recovery of human hands using convolutional networks. ACM Trans Graph. 2014. https://doi.org/10.1145/2629500.
49. Shotton J, Fitzgibbon A, Cook M, et al. Real-time human pose recognition in parts from single depth images. In: Proc IEEE Comput Soc Conf Comput Vis Pattern Recognit 2011; 1297–1304
50. Zhang F, Bazarevsky V, Vakunov A, et al. MediaPipe hands: on-device real-time hand tracking. https://doi.org/10.48550/arxiv.2006.10214
51. Zhang J, Li W, Ogunbona PO, et al. RGB-D-based action recognition datasets: a survey. Pattern Recognit. 2016;60:86–105.
52. Kim J-W, Choi J-Y, Ha E-J, Choi J-H. Human pose estimation using mediapipe pose and optimization method based on a humanoid model. Appl Sci. 2023;13(4):2700.
53. Borboni A, Roberto P, Sandrini S, Carbone G, Pellegrini N. Role of reference frames for a safe human–robot interaction. Sensors. 2023;12:5762.
54. Shaikh MB, Chai D. RGB-D data-based action recognition: a review. Sensors. 2021;21:4246.
55. Li J, Yu Q, Xu H, et al. Measuring and modeling human bodies with a novel relocatable mechatronic sensor-net. Text Res J. 2019;89:4131–47.
56. Chen H, Zhao H, Qi B, et al. Human motion recognition based on limit learning machine. Int J Adv Robot Syst. 2020. https://doi.org/10.1177/1729881420933077.

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.